

# GO-07 04: Кодогенерация. Решение проблем рефлексии

## Описание:

Итак, в прошлых главах мы познакомились с различными механизмами в Go, которые позволят нам анализировать и генерировать код под почти любые наши задачи. Давайте о них вспомним:

- Рефлексия - способность программы анализировать свои объекты во время выполнения.
- АСТ дерева - способность программы анализировать свое АСТ дерево и изменять его (Добавлять/Удалять/Изменять ноды дерева).
- Templates - способность программы генерировать любой текст на основе шаблонов со специальным синтаксисом для вставок данных из кода.

Теперь давайте посмотрим, как мы можем применять это на практике.

## Полезные ссылки:

- [Рефлексия в Go: плюсы и минусы](#)
- [Golang Templates Cheatsheet](#)
- [easyjson Github - библиотека на основе кодогенерации](#)

## Задание:

В первом задании мы с вами генерировали мапу из структуры и обратно при помощи пакета reflect. Помимо того что мы используем пакет, который может ударить по производительности приложения, мы завязываем логику статически типизированного языка на то, что типы теперь не статичны, мы их заранее не знаем и вообще обрабатывать функция может все, что угодно. Это накладывает свои сложности на разработку приложения. В этом задании мы предлагаем вам проверить тот же трюк (конвертация структуры в мапу), только при помощи кодогенерации.

Что нужно сделать:

- Нужно дописать темплейт в файле `module07/assets/template/marshaller.gotmpl`. Темплейт должен генерировать метод `StructToMap`, который конвертирует структуру `Config` из пакета `module07/internal/config` в `map[string]interface{}`.
- Сгенерировать код маршаллера по темплейту. Для того чтобы запустить генерацию, нужно раскомментировать функцию `Task04()` в функции `main` пакета `module07/cmd/app` и запустить команду `make run`.
- Написать 2 бенчмарка в файле `module07/internal/config/marshaller_bench_test.go`:

- Первый бенчмарк запускает функцию `converter.StructToMap` из первого задания, в качестве параметра выступает конфиг из пакета `module07/internal/config`.
- Второй бенчмарк запускает сгенерированный метод `StructToMap`, параметров у этого метода быть не должно, так как это метод конкретной структуры `Config`.
- Запустите и посмотрите результаты бенчмарка при помощи команды `make bench_04` или же самостоятельно (если сравните бенчмарки через утилиту `benchstat` - это будет дополнительным плюсом!).

Условия:

- Генерацию значений для темплейта (такие как имена полей, например) в этот раз попробуйте получить на основе анализа AST дерева.

Порядок действий:

1. В вашем проекте `module07` сделайте новую ветку `module07_04`.
2. В пакете `module07/assets/template` дополните темплейт `marshaller.gotmpl`.
3. Сгенерируйте маршаллер при помощи команды `make run`.
4. Создайте 2 бенчмарка в файле `module07/internal/config/marshaller_bench_test.go`.
5. Запустите бенчмарки при помощи команды `make bench_04` и проанализируйте результат (разницу в производительности).
6. В качестве ответа пришлите ссылку на `merge request` в ветку `master` вашего проекта ветки `module07_04`.